determining whether the request hits a tag stored in the cache, and

if so, generating a single external transaction to read the requested data into the agent.

21. The method of claim 20, wherein the second determining step includes:

comparing address information of the data request with tags stored in the internal cache, and

identifying a tag hit when the address information matches a stored tag.

22. An agent comprising a transaction queue, the transaction queue further comprising a plurality of queue entries, each queue entry comprising:

an address field,

a first status field to store data associated with a first transaction based on the address field, and

another status field to store data associated with another transaction based on the address field.

23. A method of processing a data request within a processing agent comprising:

posting the data request internally within the agent,

determining whether the request hit the cache,

when the request misses the cache, posting a series of external transactions to fill a cache line with data associated with the data request, the external transactions directed to a data-line-sized data item identified by an address of the data request and to at least one other data-line-sized data item adjacent to the first data item.

REMARKS

Claims 1-23 remain pending in the application. Applicants thank the Examiner for the courtesy of the March 27, 2001 interview. In view of the foregoing amendments and following remarks, Applicants respectfully request allowance of the application.

THE 35 U.S.C. § 112 REJECTIONS ARE TRAVERSED BY AMENDMENT

Claims 1 and 11 are subject to 35 U.S.C. § 112, second paragraph, objecting to use of "adapted to" as indefinite. At the Examiner's suggestion, Applicants traverse this rejection by amendment. No surrender of subject matter is intended by any amendment made by this filing.

Applicant requests that the foregoing amendments be entered even if it is deemed not to place the case in condition for allowance because the amendment will reduce issues for appeal.

THE 35 U.S.C. § 102 REJECTIONS SHOULD BE WITHDRAWN

Claims 1-23 stand rejected as anticipated by Chittor, U.S.P. 6,061,764 ("Chittor"). An anticipation rejection may be maintained only if a single prior art reference teaches each and every element of a pending claim. MPEP 2131. These rejections should be withdrawn because Chittor does not disclose all elements of the pending claims.

Claims 1 and 11 are not Anticipated by Chittor

Chittor does not teach or suggest a processing agent comprising an internal cache having a plurality of cache entries, *each entry sized to store multiple data line lengths of data*. In claim 1, a data line is the maximum amount of data that may be transferred in a single external bus transaction.

To support the rejection, the Office Action refers to cache 14 of agent 10 and argues that it stores multiple data line lengths of data. Of Chittor's entire disclosure, there are only two sentences that discuss the cache 14. They read:

> The MIOC cooperates with other agents to exchange data between the memory 200 and data caches of the agents, such as cache 14. [Col. 1:34-36]
>
> An agent 10 may modify data at an address and store the modified data in an internal cache 14 without notifying other agents on the pipelined bus 60. [Col. 2:56-58.]

Chittor provides no disclosure regarding the size or architecture of the cache 14. He says nothing that discusses the size of any cache entries therein in comparison to the maximum size of data transfers that occur on the external bus 60. Chittor has no disclosure that corresponds to the element recited in claim 1, that cache entries are sized to store multiple data line lengths of data.

These arguments were raised in Applicant's response to the first Office Action but were dismissed in the final rejection. In the final rejection, in response to Applicants' arguments regarding claim 1, the Examiner responded:

> [T]he cache lines of Chittor has to include multiple of entries that sized are store multiple data line length of data since the Chittor invention is directed to variable line length read requests from the cache lines wherein the size of a cache line

typically relates to the largest increment of data that may be transferred in a single pipelined bus transaction.

The Office Action's analysis in this regard contains several errors. Chittor discloses a computer system employing *two different buses* 60, 70 with different bus protocols. A first bus 60 is a pipelined bus; there is a predetermined maximum amount of data (called a "cache line" in Chittor) that may be transferred in a single pipelined bus transaction. If an agent 10 on the pipelined bus 60 requires use of data from multiple cache lines, it must issue multiple pipelined bus transactions to read them. See, Chittor, Col. 34-50. The second bus 70 is a serial bus. Serial bus transactions are not limited by the size of the cache lines in the same way that pipelined bus transactions are limited. See, Chittor, Col. 22-29. Agents 10-40 are disclosed as coupled to the pipelined bus 60 only; they do not generate serial bus transactions over the serial bus 70. Agent 50, however, is coupled to both the pipelined bus 60 and the serial bus.

Although Chittor's agent 10 contains a cache 14, the agent 10 is connected only to the pipelined bus 60. Therefore, the agent 10 (and its cache 14) has no interaction with the serial bus 70 where the variable sized transactions can occur. Any argument that the cache 14 has cache entry sizes that are multiples of the cache line length is guesswork; it has no support in Chittor's disclosure.

Of course, Chittor's agent 50 communicates over the serial bus 70 and, therefore, processes serial bus transactions. Chittor expressly discloses the structure that supports its variable-sized transactions on the serial bus 70. In FIG. 2, Chittor discloses a staging buffer 160 that stores data to be output on the serial bus. Chittor's disclosure is explicit regarding the width of these queue entries 162 -- they have the width of a cache line. See, Chittor, Col. 4:35-37. Thus, even the agents that support variable length transactions have cache entries whose size matches the cache line lengths defined for the pipelined bus 60. Because none of Chittor's agents 10-50 have a cache with entries that are sized to be multiples of the length of a cache line, Chittor does not disclose an agent comprising an internal cache having a plurality of cache entries, each entry sized to store multiple data line lengths of data. The § 102 rejection to claim 1, therefore, should be withdrawn.

Claim 11 recites an architecture for a processing agent that comprises an internal cache having cache entries each sized to store multiple data lines, and a transaction queue system to post external transactions, each external transaction related to a single data line. It also recites that the transaction queue system and the cache each receive data requests on a common input. Chittor discloses none of this subject matter. As noted above, no cache within any of

Chittor's agents are disclose as having a width that is sized to store multiple data lines. This is reason enough to withdraw the anticipation rejection to claim 11. Additionally, however, none of Chittor's agents are disclosed as having a cache and a transaction queue that receive data requests via a common input. This is a second basis on which to withdraw the anticipation rejection to claim 11. Chittor does not anticipate claim 11.

Claims 8 and 22 are not Anticipated by Chittor

Claims 8 and 22 define an architecture for a transaction queue in an agent. They read:

> 8. (Twice Amended) A processing agent, comprising a transaction queue having a plurality of a queue entries, the queue entries each further comprising:
>
> a primary sub entry including an address portion and status portion, the status portion provided for a first external transaction of the agent, and
>
> a secondary sub entry including a status portion provided for a second external transaction.
>
> 22. An agent comprising a transaction queue, the transaction queue further comprising a plurality of queue entries, each queue entry comprising:
>
> an address field,
>
> a first status field to store data associated with a first transaction based on the address field, and
>
> another status field to store data associated with another transaction based on the address field.

Chittor does not disclose all elements of these claims and, therefore, cannot anticipate them.

To support its rejection to claims 8 and 22, the Office Action cites to Chittor's disclosure of a splitter 140 and bus request generator 122. Chittor's description, however, is not so detailed to describe the architecture of the splitter 140 or the bus request generator 122. Chittor does not describe the architecture of either the splitter 140 of the bus request generator 122, for example, whether they store transaction data of an atomic sequence of transactions in multiple entries or in multiple fields of a single entry. From Chittor's silence the Examiner conveniently assumes, without supporting disclosure, that the architecture of the splitter 140 and bus request generator 122 are the same as recited in these claims. This is impermissible. The § 102 rejection should be withdrawn because it is based on speculation rather than express disclosure from any prior art reference.

These issues were raised to the Examiner in response to the first Office Action. The Examiner maintained the rejections based on Chittor and quoted heavily from Chittor's discussion regarding the operation of the splitter 140 and the bus request generator 122. None

of Chittor's discussion, however, describes the architecture of any entry therein. Because Chittor does not describe the architecture of entries within any transaction queue in his system, he cannot anticipate claims 8 or 22.

Claims 17 and 23 are not Anticipated by Chittor

Claim 17 recites a cache management method in which a data request is posted against a cache and, in response to a cache miss, *a sequence of external transactions* are posted to fill a cache line with data associated with the data request. The Office Action rejects claim 17 as encompassing the same scope of earlier-rejected apparatus claims but being drafted as a method claim. Such a terse rejection makes it difficult for Applicants to understand the structure relied upon for this rejection -- is it cache 14 from agent 10? is it the splitter 140 and bus request generator 122 from agent 50? -- but, either way, Chittor fails to disclose all elements of claim 17.

Applicants have already quoted above Chittor's entire disclosure with regard to the cache 14. Chittor says nothing about data requests and nothing about the agent's 10 response to any cache miss. Chittor certainly does not disclose a sequence of external transactions being posted to fill a cache line in response to any miss with regard to claim 14.

Chittor's agent 50 has no cache. It has an outbound data buffer 162 that stores data to be transferred over the serial bus 70. But Chittor's data buffer is not used as a cache would be used. There is no disclosure of any data request being applied to the data buffer to determine a hit or miss. There certainly is no disclosure of any external bus transaction being posted on a bus (much less a sequence of bus transactions) in response to hit/miss against the data buffer 162. Accordingly, Chittor has no disclosure corresponding to the subject matter of claim 17.

The Office Action impliedly recognizes the deficiencies of its rejection to claim 17. In response to Applicants' arguments of patentability, rather than cite to disclosure within Chittor that substantiates the rejection, the Office Action instead takes "official notice" that the elements of claim 17 are notoriously well-known in the art. Regardless of whether such principles are found in other prior art, the fact remains that Chittor does not anticipate claim 17. It fails to disclose all elements of claim 17. Accordingly, the § 102 rejection based on Chittor must be withdrawn.

Claim 23 recites subject matter that is similar to claim 17. When a data request misses an internal cache, claim 23 recites that a series of external transactions are posted to fill a cache line with data associated with the data request, the external transactions directed to a data-line-sized data item identified by an address of the data request and to at least one other data-line-sized data item adjacent to the first data item. This claim recites not only that a series of data transactions fill a single cache line but also that each of the transactions are directed to a data-line sized data item. As noted above with respect to claim 17, Chittor does not disclose this subject matter. In the final Office Action, Official Notice was taken that such elements are notoriously well-known in the art, an allegation that Applicants dispute. Again, even if other art can be produced that supports a rejection to claim 23, Chittor does not disclose every element of claim 23. The rejection to claim 23 should be withdrawn.

The Examiner Is Requested To Produce Documentation Supporting The Official Notice Taken In The Final Rejection.

In the final rejection, the Examiner took Official Notice that the subject matter of claims 17 and 23 are well-known in the art. Applicants dispute this and request that the Examiner cite to documentary evidence of such. Following the March 27th interview, Examiner Thai identified "The Cache Memory Book," by Jim Handy, pp. 78-81 as supporting the Official Notice taken in the final rejection. The Handy reference does not teach or suggest all elements of these claims and, therefore, provides insufficient support for the Official Notice taken in the final rejection.

Handy does not support an anticipation rejection to claims 17 or 23. Claim 17 recites posting a sequence of external transactions to fill a cache line with data associated with the data request. Claim 23 recites posting a series of external transactions to fill a cache line with data associated with the data request, the external transactions directed to a data-line-sized data item identified by an address of the data request and to at least one other data-line-sized data item adjacent to the first data item. Nothing in the Handy reference teaches this subject matter.

In § 2.2.6, the Handy reference discusses the use of write buffers during reads and writes of data. As described by Handy, a write buffer stores data evicted from a cache to make room for new data to be read into the cache. Handy describes that, without the write buffer, a write cycle would have to occur before a read of data. With a write buffer, data may be evicted to the write buffer but remain within a processor and new data may be read to the cache. The write of data from the write buffer to main memory may occur at a more leisurely pace. Handy's

pages 80-81 describe different implementations of data write-backs whose details are immaterial for this response.

The Examiner apparently believes that, because Handy discloses that a cache miss ultimate may cause a processor to issue multiple bus transactions, Handy anticipates claim 17 and 23. Not true. As recited in claim 17, the sequence of bus transactions must be done to fill a cache line. As described by Handy, only one of the transactions, the read of data, fills the cache line. The other transaction, the write of data, returns data from the write buffer to main memory. This disclosure does not satisfy the requirement of claim 17 that of posting a sequence of external bus transactions to fill a cache line. Thus, the documentary evidence provided in support of the Official Notice is insufficient to anticipate claim 17.

Handy also fails to anticipate claim 23. Not only does claim 23 recite posting a series of external transactions to fill a cache line, it also recites that one of the transactions is directed to a first data item identified by an address of the data request and another of the transactions is directed to a second data item adjacent to the first data item. Handy provides no disclosure corresponding to this claim element. Thus, Handy cannot anticipate claim 23.

Applicant respectfully submits that the Handy reference, provided by the Examiner to support the Official Notice taken in the final rejection, is insufficient to support an anticipation rejection. Accordingly, Applicants respectfully request withdrawal of any anticipation rejection to claims 17 and 23 based on the Official Notice of prior art.

CONCLUSION

All independent claims are allowable over the cited art. By extension, all claims are allowable. Further, the outstanding indefiniteness rejection has been traversed. Accordingly, Applicant respectfully requests withdrawal of all outstanding rejections and allowance of the claims.

Respectfully submitted,

Date: March 29, 2001

Robert L. Hails, Jr.
Registration No. 39,702

KENYON & KENYON

1500 K Street, N.W.
Washington, D.C. 20005
Ph.: (202) 220-4200
Fax.: (202) 220-4201

APPENDIX

1. (Twice amended) A processing agent to transfer data of a predetermined data line length in an external transaction, the agent comprising an internal cache having a plurality of cache entries, each entry sized to store multiple data line lengths of data.

2. The processing agent of claim 1, wherein the cache entries include a tag portion adapted to store address information.

3. The processing agent of claim 2, wherein the internal cache further comprises
match detection logic for the tag portions, and
control logic provided in communication with the match detection logic.

4. The processing agent of claim 1, wherein the cache entries include a cache coherency state field in association with each data line length of data.

5. The agent of claim 1, further comprising a transaction queue having a plurality of queue entries, the queue entries including a primary entry adapted to store address information and status information of a first external transaction and a secondary entry adapted to store status information of a second external transaction.

6. The agent of claim 4, wherein the status information of the first external transaction includes a field representing whether the first external transaction is part of a multiple transaction sequence.

7. The agent of claim 4, wherein the total number of primary and secondary entries equals the multiple number of data line lengths provided in the cache entries.

8. A processing agent, comprising a transaction queue having a plurality of a queue entries, the queue entries each further comprising:

a primary sub entry including an address portion and status portion, the status portion provided for a first external transaction of the agent, and

a secondary sub entry including a status portion provided for a second external transaction.

9. The transaction queue of claim 8, wherein the status portion of the primary entry includes a field representing whether the first transaction is part of a multiple transaction sequence.

10. The transaction queue of claim 8, further comprising control logic adapted to cycle through the queue entries and post transactions therefrom.

11. (Twice Amended) A processing agent, comprising:

an internal cache having cache entries each sized to store multiple data lines, and

a transaction queue system to post external transactions, each external transaction related to a single data line,

wherein the internal cache and the transaction queue system each receive data requests on a common input.

12. The processing agent of claim 11, wherein the internal cache and the transaction queue system communicate by signal lines.

13. The processing agent of claim 12, wherein the signals lines include a cache hit signal line and a tag hit signal line.

14. The processing agent of claim 11, wherein the transaction queue system comprises a plurality of queue entries, each queue entry comprising:

a primary entry including an address portion and status portion, the status portion provided for a first external transaction of the agent, and

a secondary entry including a status portion provided for a second external transaction.

15. The transaction queue of claim 14, wherein the status portion of the primary entry includes a field representing whether the first transaction is part of a multiple transaction sequence.

16. The transaction queue of claim 14, further comprising control logic adapted to cycle through the queue entries and post transactions therefrom.

17. A method of processing a data request within a processing agent comprising:

posting the data request internally within the agent,

determining whether the request hit the cache,

when the request misses the cache, posting a sequence of external transactions to fill a cache line with data associated with the data request.

18. The method of claim 17, wherein the determining step includes:

comparing address information of the data request with tags stored in the internal cache, and

identifying a cache miss when the address information does not match any stored tag.

19. The method of claim 18, wherein the determining step further includes:

when address information matches a stored tag, reading cache coherency state information associated with the requested data, and

identifying a cache miss when the cache coherency state information is invalid for a request type of the data request.

20. The method of claim 17, further comprising, when the request hits the cache:

determining whether the request hits a tag stored in the cache, and

if so, generating a single external transaction to read the requested data into the agent.

21. The method of claim 20, wherein the second determining step includes:

comparing address information of the data request with tags stored in the internal cache, and

identifying a tag hit when the address information matches a stored tag.

22. An agent comprising a transaction queue, the transaction queue further comprising a plurality of queue entries, each queue entry comprising:

an address field,

a first status field to store data associated with a first transaction based on the address field, and

another status field to store data associated with another transaction based on the address field.

23. A method of processing a data request within a processing agent comprising:

posting the data request internally within the agent,

determining whether the request hit the cache,

when the request misses the cache, posting a series of external transactions to fill a cache line with data associated with the data request, the external transactions directed to a data-line-sized data item identified by an address of the data request and to at least one other data-line-sized data item adjacent to the first data item.